

User Manual of the modeling tools

Table of Contents

1. Introduction.....	1
1.1 What the tools do for you.....	1
1.2 Prerequisites.....	2
1.2.1 Java virtual machine.....	2
1.2.2 Java virtual machine in Matlab.....	2
2. Installation.....	2
2.1 Installation under Linux.....	2
2.1.1 Base installation.....	2
2.1.2 Adding Matlab support.....	4
2.2 Installation under Windows.....	5
2.2.2 Adding Matlab support.....	5
3. Use.....	5
3.1 Command line use (Linux or Windows), without Matlab.....	5
3.2 Use from Matlab.....	6
4. Notes on extension/correction of the program.....	6
5. Troubleshooting.....	7
5.1 Common problems and their solutions:.....	7

1. Introduction

1.1 What the tools do for you

The different tools present in this set of tools allow you to:

- Transforms a file written in the TSed language into SBML
- Transforms a file written in SBML into a file in Matlab format containing the ODE system, and also generate an additional custom file with parameters values for the ODE system
- Transforms a file written in SBML into a LaTeX file
- Visualize a file written in SBML with a graph

These various tools can be used from the command line of Linux or Windows, they can also be accessed from Matlab. Although some file are generated in Matlab format, it is very likely that other free tools (like scylab or Octave) which uses a format compatible or similar to Matlab, might also be able to use these files.

For details on installation see section 2 and for use see section 3.

1.2 Prerequisites

1.2.1 Java virtual machine

In order to run the tools, you must have a working Java virtual machine, under Linux you can test it by typing “java” in a command line which should output:

```
Usage: java [-options] class [args...]  
          (to execute a class)  
or  java [-options] -jar jarfile [args...]  
          (to execute a jar file)  
...
```

In order to run most commands of this set of programs you must have at least Java version 1.4, and to interpret TSed files you need version 1.5 or more. You can know the version of Java installed on your computer by typing “java -version”.

If your version of Java is too old you can download a newer version on <http://java.sun.com>

1.2.2 Java virtual machine in Matlab

The same condition apply if you want to run the programs from Matlab. As Matlab has its own virtual machine, the version from your system and the one in Matlab may be different. You can know the version of the virtual machine of Matlab by typing “version -java” in a Matlab window.

If the version of Java in Matlab is out-dated you can force Matlab to use another virtual machine, by setting the “MATLAB_JAVA” environment variable to point to the new virtual machine.

For example, if you are using a C shell and your virtual machine is installed in “/usr/lib/j2sdk1.5-sun”, you can run:

```
setenv MATLAB_JAVA /usr/lib/j2sdk1.5-sun/jre
```

Please read the Matlab documentation for details.

2. Installation

2.1 Installation under Linux

2.1.1 Base installation

After the basic installation you should be able to launch the any program from the command line, see “Use under Linux” for details.

1. First unpack the file “modeling.tar.gz” in the directory of your choice:

```
tar -xvzf modeling.tar.gz
```

2. Then configure your LD_LIBRARY_PATH environment variable to point to the “lib/linux” subdirectory of the newly created directory.

- If you are using a bash shell, modify the file name “.bashrc”, by adding the line:

```
export LD_LIBRARY_PATH=/<path>/lib/linux
```

- If you are using a C shell or a TC shell, modify the file named “.cshrc”, by adding this line:

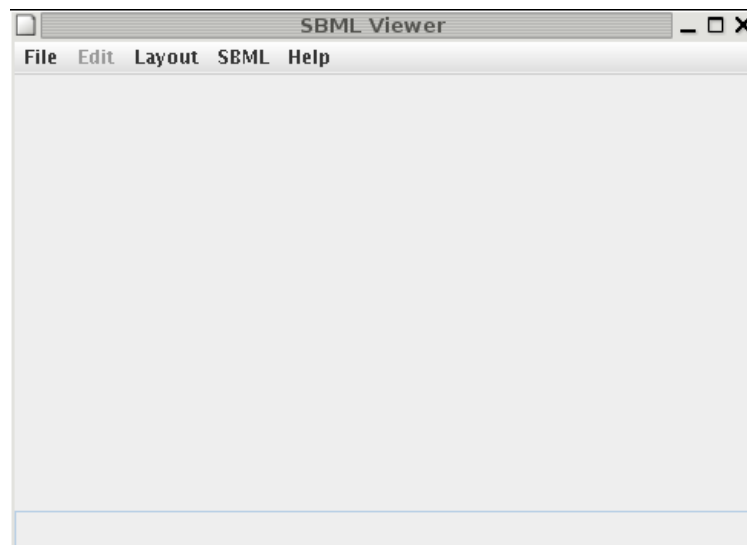
```
setenv LD_LIBRARY_PATH /<path>/lib/linux
```

where “<path>” is the directory you created at step 1.

3. You need to restart your shell to have this change applied, you can check that it is correct by looking at the output of the “env” command.
4. To test your installation, you can now run SView for the first time, go to the subdirectory created at the first step, and run “./SView”.

Note: on some systems, a window telling you that the library wasn't found could appear, it will then invite you to locate “libsbmlj.so”, which is in the “lib/linux” subdirectory (note that it is independent from step 2.).

You should now have the SView window visible, which should look like the following picture.



2.1.2 Adding Matlab support

In order to use the different commands from Matlab, you will have to add these commands to Matlab's path, for this follow the following steps:

1. The second step is to create/modify Matlab startup file. This file contains all commands that Matlab will execute on startup.

This file is located in the “matlab” subdirectory of your home and is named “startup.m”, for example, if your home folder is “/home/dupont”, the startup file is “/home/dupont/matlab/startup.m”

Note: you can copy the sample “statup.m” file from the “matlab” subdirectory of your installation.

Your “statup.m” should look like:

```
install = '<replace-this>';  
addpath(strcat(install, '/matlab/modeling'));  
javaaddpath(strcat(install, '/lib/linux/libsbmlj.jar'));  
javaaddpath(strcat(install, '/lib/jgraph.jar'));  
javaaddpath(strcat(install, '/lib/jgraphpad.jar'));  
javaaddpath(strcat(install, '/lib/jigloo.jar'));  
javaaddpath(strcat(install, '/modeling.jar'));
```

You have to replace “<replace-this>” by the installation directory.

2. You can now test your installation by running Matlab, and then type “SView” to run the SView program.

2.2 Installation under Windows

Simply unzip the “modeling_windows.zip” file in the directory of your choice. Then you can immediately test by double clicking the “SView.bat” file.

2.2.2 Adding Matlab support

This can be done exactly in the same way than section 2.1.2 except that the slashes are back-slashes.

3. Use

3.1 Command line use (Linux or Windows), without Matlab

- Converting a TSed file into SBML, is done with the following command:

```
tsed2sbml <input> <output>
```

the <output> is optional.

Example:

```
tsed2sbml myFile.txt toto.xml
```

will transform the TSed file named “myFile.txt” into the SBML file toto.xml.
- Converting a SBML file into Matlab files, is done with the following command:

```
sbml2matlab <input> <output>
```

the <output> is optional.

Example:

```
tsed2sbml myFile.xml toto.m
```

will transform the SBML file named “myFile.xml” into a Matlab file “toto.m”, plus another Matlab file “toto_simulate.m”, that contains basic simulation commands..
- Converting a SBML file into LaTeX files, is done with the following command:

```
sbml2latex <input> <output>
```

the <output> is optional.

Example:

```
tsed2sbml myFile.xml toto.tex
```

will transform the SBML file named “myFile.xml” into the LaTeX file

“toto.tex”, plus a shell script named “toto_build_latex.sh” that, if run, would build a PDF file from the LaTeX file.

- Starting SView is simply done by running “SView”, for additional informations on SView please report to SView's user manual. Note that all the previous commands can also be accessed through SView.

3.2 Use from Matlab

The use from Matlab is exactly the same as the command line use under Linux, except that the names have to be between apostrophe, because otherwise Matlab considers them as variables.

Example:

```
sbml2matlab 'Test2.xml'
```

4. Notes on extension/correction of the program

This source code of this program is available in the `source.tar.gz` archive.

Once unpacked, you can import it in the Eclipse environment were it was designed. The code is heavily documented and very self-explanatory, but here are a few tips:

- The MathML parsing is done through the object MathML2Matlab and MathML2Latex, both classes inherits from MathML2a. If you want to export MathML into a new kind of language you should create a new classe inheriting from MathML2a.
- The parsing of the SBML itself is done by the libSBML library, the full JavaDoc is available on the SBML website.
- The analysis of the parsed SBML is done by the classes AnalyzedSomething, where “Something” is a SBML model, a species or a reaction.
- For the packaging two shell script are available “build_linux.sh” and “build_windows.sh”. They create the “modeling_linux.tar.gz” and “modeling_windows.zip” files respectively.
- The “TSed” and “SView” part are in the ext-bin subfolder, so that they are not mixed with the rest of the code.
- This is also the case of the tests which are in the “unittests” sub folder.
- Concerning the SBML library for Linux, it was compiled with the “--with-expat --with-java” parameters. For the Windows version of the library, the expat version of the library was tested, but did not work, so that the xerces version was finally used.

5. Troubleshooting

5.1 Common problems and their solutions:

- The program SView does not quit correctly, after closing the window, the shell is still occupied.

Solution: force quit by typing “Ctrl + C”